Expert Systems with Applications 39 (2012) 6435–6445

Contents lists available at SciVerse ScienceDirect

Expert Systems with Applications

journal homepage: www.elsevier.com/locate/eswa

Multiobjective heuristic search in road maps ☆

E. Machuca *, L. Mandow **

Dpto. Lenguajes y Ciencias de la Computación, Universidad de Málaga, Boulevar Louis Pasteur, 35, Campus de Teatinos, 29071 Málaga, Spain

ARTICLE INFO

Keywords:
Multiobjective shortest path problem
Best-first search
Heuristic search
Artificial intelligence
Road networks

ABSTRACT

This article considers the application of exact multiobjective techniques to search in large size realistic road maps. In particular, the NAMOA* algorithm is successfully applied to several road networks from the DIMACS shortest path implementation challenge with two objectives. An efficient heuristic function previously proposed by Tung and Chew is evaluated. Heuristic values are precalculated with search. The precalculation effort is shown to pay off during the multiobjective search stage. An improvement to the calculation procedure is also proposed, resulting in added improved time performance in many problem instances.

1. Introduction

Real decision problems frequently involve the consideration of multiple conflicting objectives. Solution to these problems normally results in a set of so-called Pareto optimal or nondominated solutions. These define the optimal tradeoffs between the objectives under consideration.

The multiobjective shortest path problem is an extension of the shortest path problem with several objectives. Multiobjective problems are more complex computationally than their scalar counterparts. In fact, a number of objections question the application of exact multiobjective heuristic search techniques to large size realistic problems. In the first place, the number of solutions to these problems is known to grow exponentially with goal depth in the worst case, even for the two-objective case. The problem is therefore formally intractable (Hansen, 1979). In the second place, multiobjective heuristic search has been regarded for some time as lacking the formal properties of its scalar counterparts. In particular, the work of Stewart and White presented MOA*, a multiobjective extension of the A* algorithm (Hart, Nilsson, & Raphael, 1968), but could not establish a significant relation between the accuracy of heuristic information and search efficiency in MOA* (Stewart & White, 1991). Finally, good heuristic functions are frequently hard to find even for a single objective. Particularly, precalculating heuristic estimates with search was formally shown to be more inefficient than blind search in certain cases (Hansson, Mayer, & Valtorta, 1992; Valtorta, 1984).

However, several recent results promise to reduce the impact of the above objections in practical applications. In the first place, a new multiobjective heuristic search algorithm (NAMOA*) has been proposed. This algorithm has been found to be optimal over the class of admissible multiobjective algorithms, and to strictly dominate MOA* (Mandow & Pérez de la Cruz, 2010). Furthermore, the efficiency of NAMOA* has been formally linked to heuristic accuracy in the sense that more informed heuristics can never decrease efficiency (Mandow & Pérez de la Cruz, 2010). Rather, efficiency normally increases with better informed heuristics, just as in the case of single-objective A*.

In the second place, several authors have pointed out that particular classes of multiobjective search problems do not exhibit exponential worst-case behavior (Mandow & Pérez de la Cruz, 2009; Müller-Hannemann & Weihe, 2006). In particular, in polynomial state spaces with bounded integer costs and a fixed number of objectives, the number of nondominated solutions can be shown to grow only polynomially with goal depth in the worst case (Mandow & Pérez de la Cruz, 2009).

Finally, pattern database heuristics (Culberson & Schaeffer, 1998) and hierarchical graph search (Holte, Perez, Zimmer, & MacDonald, 1996; Holte, Grajkowski, & Tanner, 2005) have overcome previous formal limitations, and shown that under certain circumstances heuristics precalculated with search can be used to boost search efficiency. Regarding multiobjective search, the work of Tung and Chew proposed a precalculated multiobjective heuristic (TC) that calls for solving one single-objective problem for each objective under consideration (Tung & Chew, 1992). However, this multiobjective heuristic has not received much attention and its impact in many potential practical applications still lacks adequate evaluation.

Applications of multiobjective search in road maps range from off-line planning of routes for hazardous material (hazmat) transportation (Caramia, Giordani, & Iovanella, 2010; Dell'Olmo, Gentili,

☆ This work is partially funded by Este trabajo está parcialmente financiado por: Consejería de Innovación, Ciencia y Empresa. Junta de Andalucía (España), P07-TIC-03018.

* Main corresponding author. Tel.: +34 (9) 52 132863; fax: +34 (9) 52 131397.
** Corresponding author.
E-mail addresses: machuca@lcc.uma.es (E. Machuca), lawrence@lcc.uma.es (L. Mandow).

0957-4174/$ - see front matter © 2011 Elsevier Ltd. All rights reserved.
doi:10.1016/j.eswa.2011.12.022

& Scozzari, 2005; Erkut, Tjandra, & Verter, 2007) to route search in car navigation systems (Delling, Sanders, Schultes, & Wagner, 2009; Kanoh, 2007; Kanoh & Hara, 2008; Kim, Jo, Kim, & Gen, 2009; Schultes, 2008). An overview of multiobjective routing problems can be found in Jozefowiez, Semet, and Talbi (2008). However, most previous applications of multiobjective search have involved approximation schemes or relatively small sized problems.

This paper describes the application of multiobjective heuristic search to large size realistic multiobjective road map search problems. In particular, NAMOA* with the TC heuristic is successfully applied to several road maps obtained from the DIMACS shortest path implementation challenge. The effort invested in the calculation of the TC heuristic is shown to be largely compensated by the speed up of multiobjective search. An improvement to the TC heuristic is also proposed and shown to add improved efficiency in many problem instances.

The paper is organized as follows: next section summarizes related work and previous results in single and multiobjective search. Section 3 describes the problem analyzed and section 4 the heuristic functions to be tested. Section 5 presents experimental results, which are discussed in Section 6. Finally, some conclusions and future work are outlined.

2. Antecedents

2.1. Single-objective search

The shortest path problem is probably one of the most studied problems in Operational Research. Dijkstra's algorithm (Dijkstra, 1959) is an efficient solution to the problem with many practical applications in route planning. The A* algorithm (Hart et al., 1968) is another algorithm that incorporates the idea of heuristic evaluation functions to increase search efficiency. In A* nodes are selected for expansion according to a characteristic evaluation function $f(n) = g(n) + h(n)$ that combines $g(n)$, the cost of the best known path to n, with $h(n)$, an heuristic estimate of the cost of a path reaching the goal from n. When $h(n)$ is a lower bound (i.e. optimistic or admissible), A* is guaranteed to find optimal solutions. When heuristic functions satisfy the so-called consistency property, the use of better informed heuristics can be used to speed up search. In this case, A* can also be shown to be optimal over the class of admissible algorithms (Dechter & Pearl, 1985).

Single-objective route planning in road maps is currently an active area of research. Pearl (1984) (Section 5.3) presented a formal analysis on the relative performance of Dijkstra's and the A* algorithm (using an Euclidean distance heuristic) on road maps with randomly distributed cities and uniform density. A new technique for computing distance bounds has been presented by Goldberg and Harrelson (2005). The idea is to use precalculated optimal distances to certain *landmarks* to provide improved heuristic estimates using the triangle inequality. The approach is called ALT (A* search, Landmarks, and Triangle inequality). Recently, the use of bidirectional A* in the more challenging problem of search in time-dependent road networks has also been proposed (Nannicini, Delling, Liberti, & Schultes, 2008).

Many recent works concentrate on a variety of speed up techniques for Dijkstra's algorithm. A description of these techniques is outside the scope of this article. Good overviews of this field can be found elsewhere (Delling et al., 2009; Schultes, 2008). In general, speed up techniques exploit information gathered in previous extensive searches of the road map. The challenge is to achieve fast shortest-path queries with practical preprocessing time and memory. A recent work established that the optimal adjustment in many recent techniques (for example, the assignment of landmarks to a graph in the ALT technique cited above) is NP-hard (Bauer, Columbus, Katz, Krug, & Wagner, 2010). In practice, these adjustments are frequently settled experimentally.

2.2. Multi-objective search

Research in multiobjective search dates back at least to the analyses of Hansen (1979), that included an extension of Dijkstra's algorithm to the biobjective case. Martins extended Dijkstra's algorithm to the general multiobjective case (Martins, 1984). Multiobjective problems replace the traditional scalar concept of optimal solutions with the notion of nondominated (Pareto-optimal) solutions. Nondominated solutions cannot be improved in one objective without degrading value in at least another one.

Dominance induces only a partial order relation. In consequence, in multiobjective search many different nondominated paths may reach any node. In fact, the number of such paths can be shown to grow exponentially with node depth in the worst case even with just two objectives (Hansen, 1979). The general problem is therefore formally intractable. However, several authors have noted that special classes of problems do not exhibit this worst case behavior (Mandow & Pérez de la Cruz, 2009; Müller-Hannemann & Weihe, 2006). In particular, in polynomial state spaces with bounded integer costs and a fixed number of objectives, the number of nondominated solutions can be shown to grow only polynomially with goal depth in the worst case (Mandow & Pérez de la Cruz, 2009). Since road maps are generally defined over a surface, the number of nodes typically grows quadratically with node depth in the worst case, assuming node density is locally upper bounded by some value. Therefore, as long as the number of objectives is fixed and arc costs are bounded and integer, the problem becomes formally tractable.

Multiobjective search in road maps is a relatively unexplored area when compared to single-objective search. Limited experiments with multiobjective genetic algorithms in dynamic networks for car navigation are described in Kanoh (2007), Kanoh and Hara (2008), and Kim et al. (2009). These involve search in networks with hundreds or a few thousand nodes and three or four objectives. Martins' multiobjective search algorithm has been applied to route planning of hazardous materials in networks with a few hundreds of nodes with two (Dell'Olmo et al., 2005) and three objectives (Caramia et al., 2010). A typical objective in these applications is the minimization of risk, frequently complemented with the minimization of time and/or distance. An overview of the application of multiobjective route planning to hazardous material transportation can be found in Erkut et al. (2007). Some attempts have been made to extend speed up techniques typically used with single-objective route planning to the multiobjective case. Delling and Wagner (2009) extended the single-objective SHARC (Shortcuts + ARC-flags) technique to be used with Martins blind multiobjective label-setting algorithm. Reports on the exact solution of problems with two objectives and networks of up to 77,740 nodes are presented. An approximation technique is successfully used to search much larger networks, where the extensive multiobjective preprocessing searches are not practical (Delling & Wagner, 2009).

Finally, Raith and Ehrgott compared a number of blind multiobjective search algorithms on different test cases, including modified road networks of up to 330,386 nodes and two objectives (time and distance) (Raith & Ehrgott, 2009). The analysis concluded that a two-phase method combining two multiobjective label-setting searches provided the best results in these networks.

Regarding multiobjective heuristic search, several extensions of A* to the multiobjective case have been proposed. NAMOA* has recently been shown to enjoy equivalent formal properties of A*. In particular it has been shown to be optimal over the class of admissible multiobjective search algorithms (Mandow & Pérez de la Cruz, 2010).

Tung and Chew (1992) proposed a heuristic multiobjective search algorithm using two different precalculated heuristic functions: the first one dedicated to label selection, and the second one to label pruning (TC). Paradoxically, the use of accurate heuristics for label selection was found to degrade time efficiency in multiobjective search (Machuca, Mandow, & Pérez de la Cruz, 2009). However, the use of NAMOA* with the TC pruning heuristic was found to result in important gains in efficiency.

To the author's knowledge, no previous results on the application of heuristic multiobjective search to road maps have been reported. This article tests the combination of NAMOA* with the TC heuristic on a set of random problems obtained from realistic road networks of up to 1,070,376 nodes. A more efficient calculation method for the TC heuristic is also described.

3. Road maps, problems, and cost structure

The experiments described in this article have been performed over publicly available road maps from the "9th DIMACS Implementation Challenge: Shortest Path".[1] A set of twelve road maps of increasing size was prepared as part of the Challenge.[2] These include arcs representing road segments, and nodes representing road junctions. Coordinates (longitude and latitude) are provided for each node. The data were taken from the 2000 US Census Bureau's TIGER/Line® files (Topologically Integrated Geographic Encoding and Referencing system).

Particularly, four maps from the DIMACS map set have been considered: New York city, San Francisco bay, Colorado, and Florida. The sizes for these maps are presented in Table 1. A sample rendering of the New York city map is shown in Fig. 1. A set of fifty problems have been generated for each map, selecting random start and goal nodes using a uniform distribution.

According to the information provided by the Challenge organizers, arcs are labeled with two different cost values: *physical distance* (c_1) and *travel times* (c_2). These are the two objectives to be minimized by multiobjective search, and shall be grouped in a vector cost $\vec{c} = (c_1, c_2)$. The cost of a path is calculated as the sum of the costs of its component arcs. Each arc cost is defined as follows,

- Original arc distances were calculated as the so-called *great circle* distance between its nodes. This is the shortest distance taking into account the Earth's curvature and average radius. *Physical distance* values are integers obtained from the original distances using the formula $c_1 = (int)(distance \times 10 + 0.5)$, i.e. the original distance data were multiplied by 10, and truncated after adding 0.5. This means that each arc can accumulate, in the worst case, an error of 0.5 units.
- *Travel time* values are calculated as the integer part of the original distance divided by an average speed factor that depends on road category, i.e. $c_2 = (int)(distance/factor)$. There are four such categories, obtained from the TIGER/Line data, with associated factors 1.0 (primary highway), 0.8 (primary road), 0.6 (secondary and connecting road), and 0.4 (local, neighborhood, and rural road).

The experiments described in the following sections use the NAMOA* algorithm to solve the problems over the road networks. The efficiency of this algorithm can be improved using an heuristic function $H(n)$ to estimate path costs. More precisely, $H(n)$ returns a set of heuristic cost vectors such that, for each cost vector $\vec{h}^* = \left(h_1^*, h_2^*, \ldots, h_q^*\right)$ of an actual nondominated path from n to the goal, there is a vector $\vec{h} = (h_1, h_2, \ldots, h_q) \in H(n)$ that estimates $\vec{h}^*$. If the estimates are lower bounds, i.e. $\forall i\ h_i \leqslant h_i^*$, then the algorithm is guaranteed to terminate with the set of all nondominated solutions (Mandow & Pérez de la Cruz, 2010). In general, more precise estimates result in more efficient search. Therefore, finding accurate heuristic estimates is a central issue in heuristic search.

[1] http://www.dis.uniroma1.it/challenge9/.
[2] http://www.dis.uniroma1.it/challenge9/download.shtml#benchmark.

Table 1
Road networks sizes.

Name	Location	Nodes	Arcs
NY	New York City	264,346	730,100
BAY	San Francisco Bay	321,270	794,830
COL	Colorado	435,666	1,042,400
FL	Florida	1,070,376	2,712,798

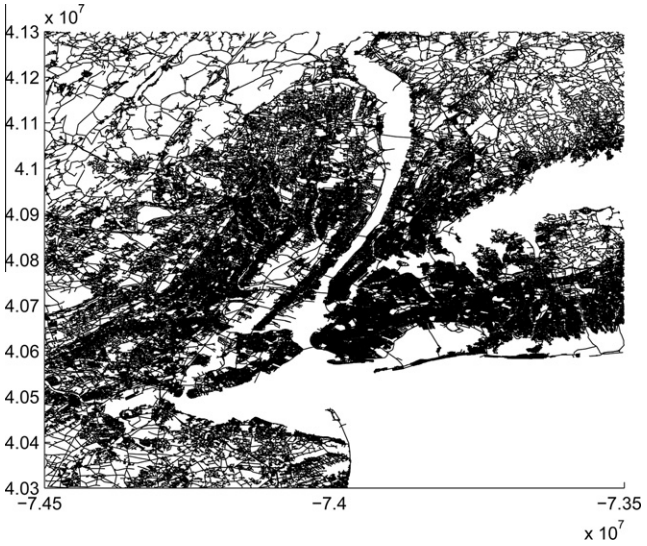

Fig. 1. Rendering of the New York City map.

4. Heuristics

Different lower bounds can be devised to estimate physical distances and travel times. All the multiobjective heuristic functions considered in this article return a single vector estimate that lower bounds *all* nondominated solution paths emanating from a given node, i.e. $H(n) = \{\vec{h}(n)\}$. Therefore, we shall refer to the multiobjective heuristic function for simplicity as $\vec{h}(n)$. The basic approach is to use $\vec{h}_0(n) = (0, 0)$ which amounts to *uninformed* or *blind* search. This section describes more informed heuristics.

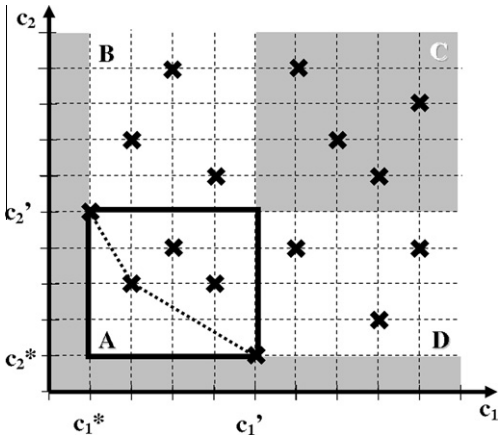

Fig. 2. A typical Pareto-front on a costs space.

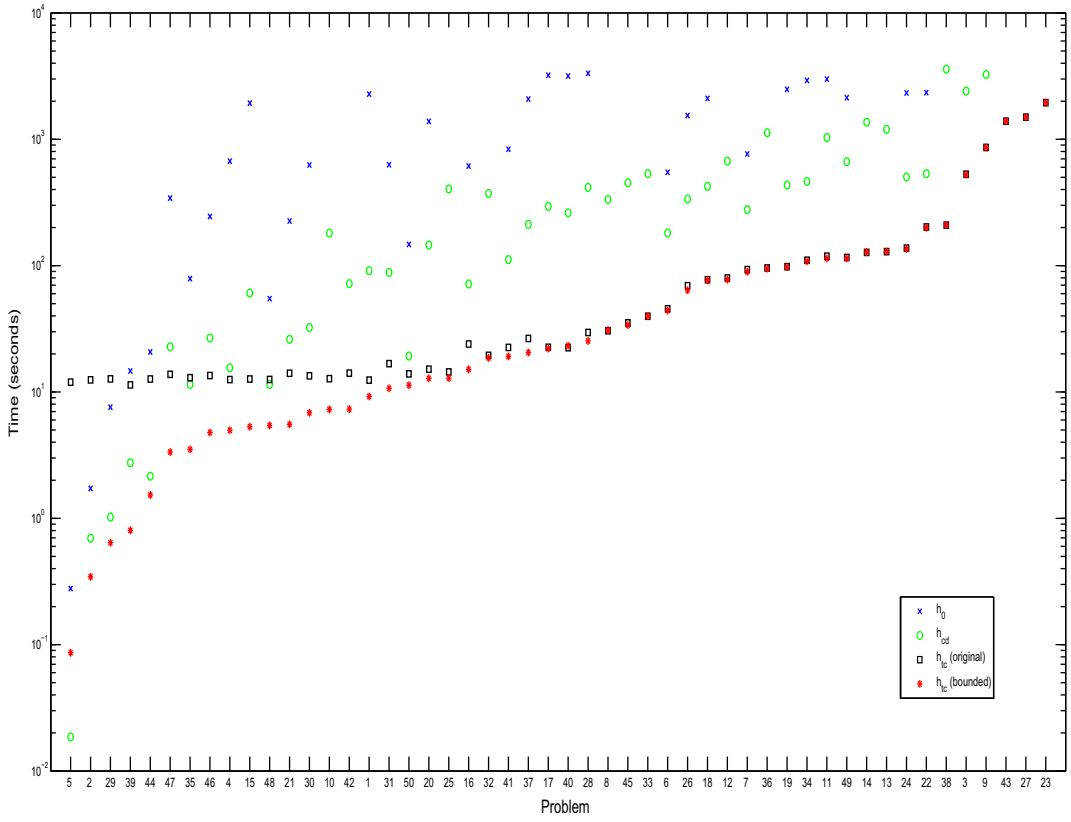

Fig. 3. Time requirements for NY map.

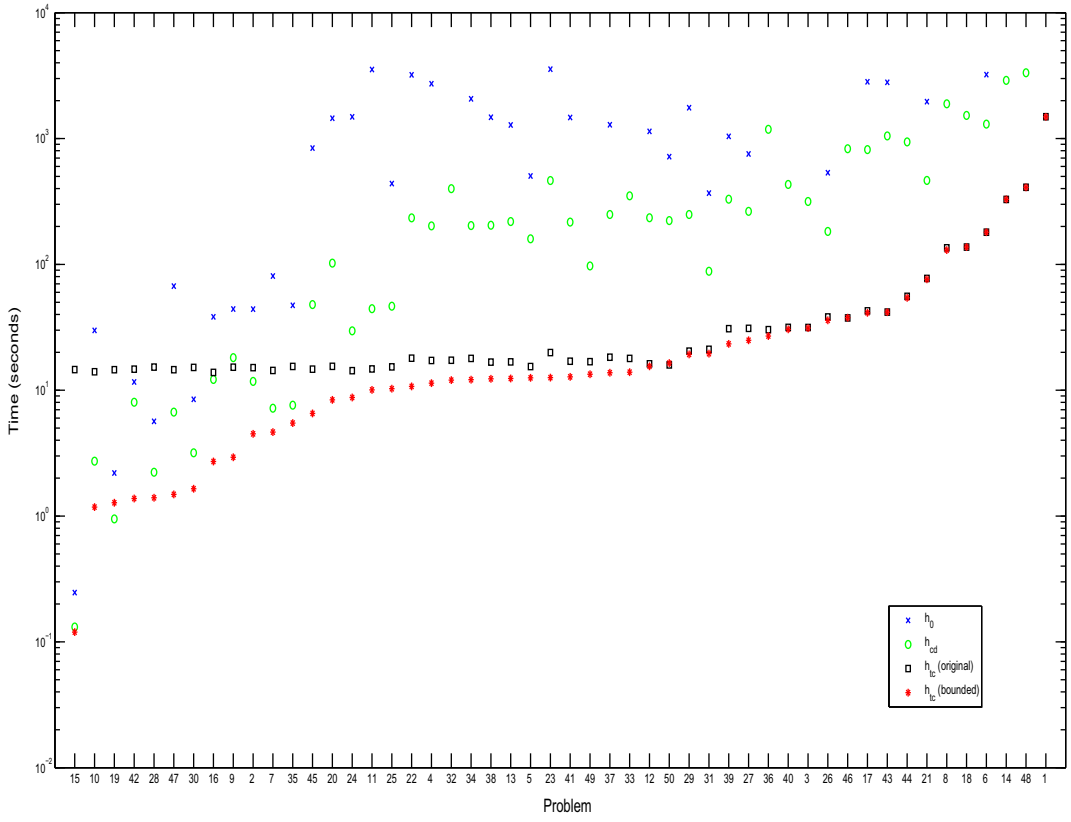

Fig. 4. Time requirements for BAY map.

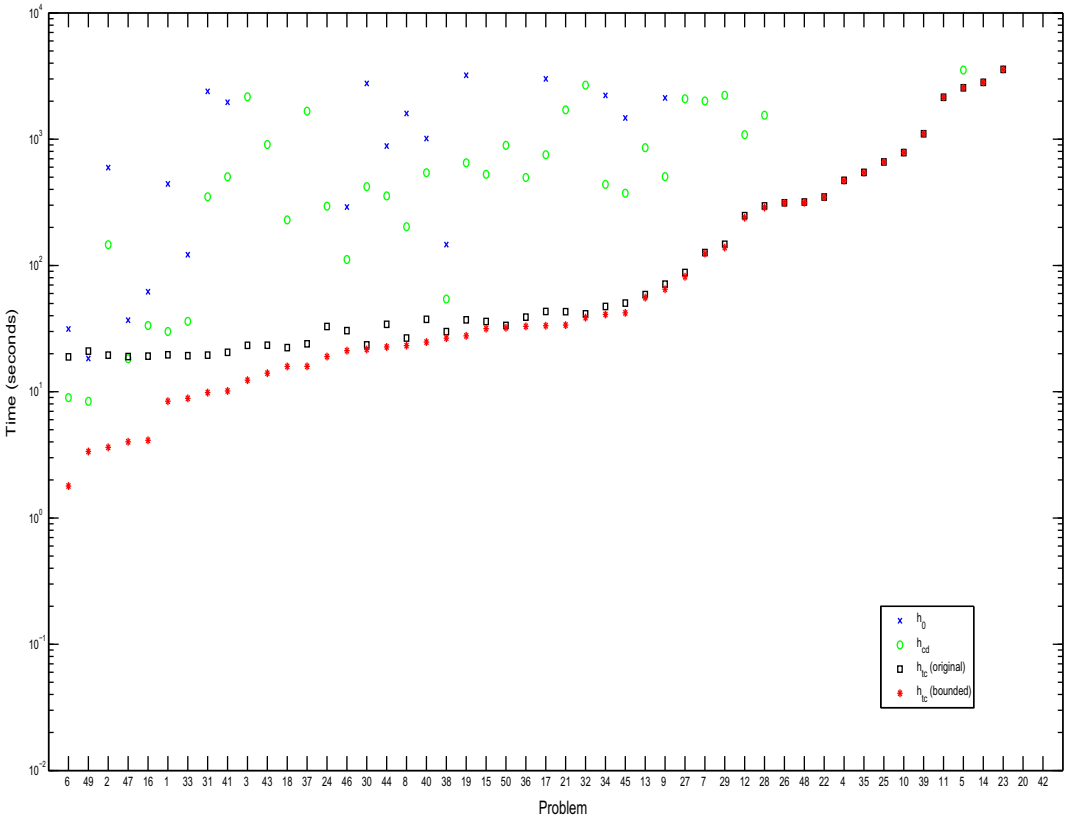

Fig. 5. Time requirements for COL map.

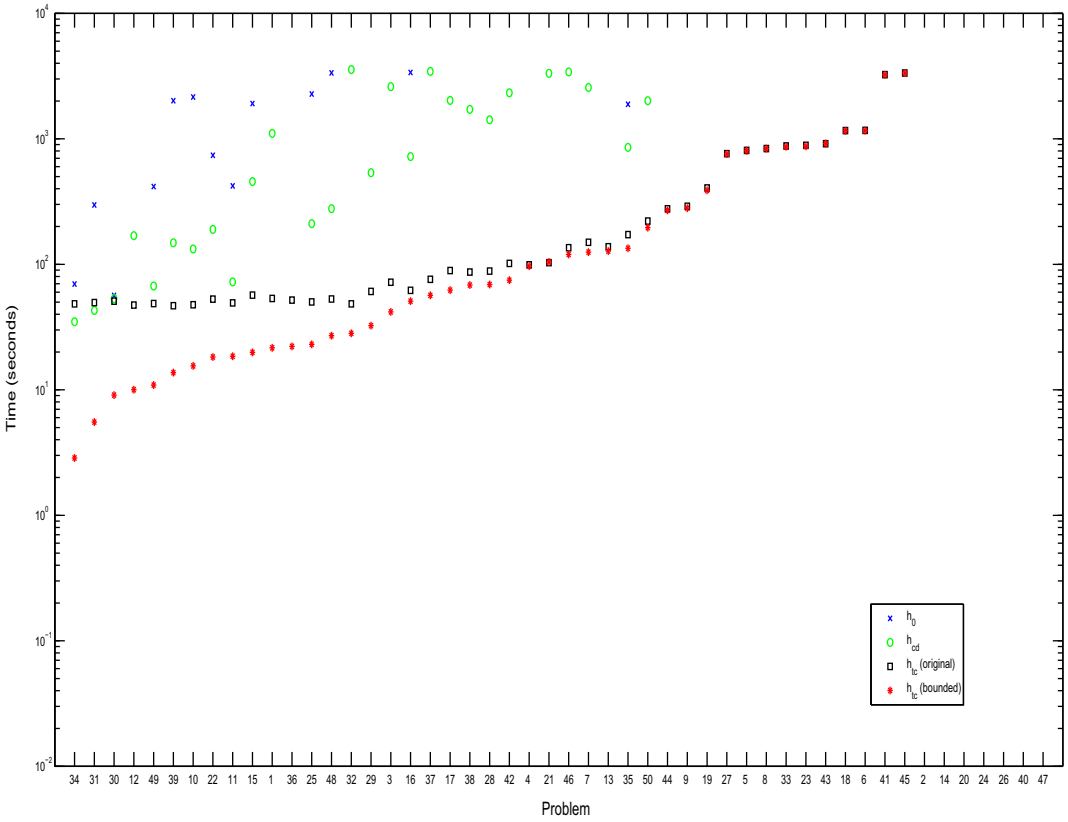

Fig. 6. Time requirements for FLA map.

4.1. Corrected great circle distance heuristic

Let us consider first an estimate for physical distance. Euclidean distance is a simple way to estimate distances between nodes in a planar surface. However, this measure presents several drawbacks for the maps considered in this study. In the first place, node positions are described through longitude and latitude, and not by coordinates in a plane. Secondly, arc distances are calculated in the maps using the *great circle* distance as explained in section 3. Finally, arc distances in the maps are truncated and paths frequently accumulate errors (up to 0.5 units per arc). Therefore, the straightforward calculation of Euclidean or great circle distances between nodes can easily overestimate optimal distances calculated using map data. This yields both measures inadmissible and inconsistent in general.

Nevertheless, great circle distances can be *corrected* subtracting the maximum possible error to obtain an admissible heuristic. Let n be some node, and γ the destination node. Let $d(n)$ be the great circle distance between n and γ, and $d^*(n)$ the actual optimal physical distance of a path joining both nodes. Let us further assume this optimal path actually joins both nodes following a circular line over the Earth's surface. Ideally, the equality $d(n) = d^*(n)$ should hold. However, due to truncations in physical distance, $d(n)$ will in general overestimate $d^*(n)$. The worst overestimate would occur if the path were made up of a large number of cocircular arcs, each one accumulating a maximum truncation error of 0.5 units.

The cumulative error in this case can be upper bounded as follows. Let us group all the arcs in the graph in disjoint categories such that each category groups all arcs with the same physical distance, and there are N_i arcs of physical distance L_i for each category i. Let us further assume that $\forall i < j \;\; L_i < L_j$. The truncation error $e(n)$ of a straight path joining n with γ is upper bounded as $e(n) = 0.5 \times N(n)$ where $N(n)$ is the maximum number of map arcs that can be concatenated to reach a distance of $d(n)$, i.e. N_1 arcs of length L_1 plus N_2 arcs of length L_2, etc. More precisely, let us define a_i as the cumulative physical distance of all arcs in the graph up to category i,

$$a_i = \sum_{j=0}^{i} N_j \times L_j \tag{1}$$

Table A.7
Data from NY map problems.

Prob.	Source	Goal	t_{h_0}	$t_{h_{cd}}$	$t_{h_{tc}}(o)$	$t_{h_{tc}}(b)$	t_{namoa}	$\lvert C^*\rvert$
1	33,502	163,335	2279.89	91.22	12.43	9.22	0.85	45
2	198,561	195,430	1.73	0.70	12.47	0.34	0.04	12
3	40,851	4310	–	2408.30	528.29	525.87	515.72	344
4	19,103	95,503	670.30	15.53	12.53	4.98	0.24	24
5	65,190	57,030	0.28	0.02	11.95	0.09	0.02	1
6	172,882	189,944	548.33	181.62	45.52	44.17	32.87	163
7	181,176	151,910	764.42	277.22	92.99	89.46	80.06	308
8	177,414	103,345	–	334.00	30.56	30.72	18.66	122
9	186,166	71,968	–	3252.33	862.23	862.22	850.69	487
10	50,616	76,333	–	180.79	12.75	7.27	1.04	31
11	56,699	159,358	2992.14	1031.99	118.74	114.02	106.95	401
12	103,987	175,817	–	672.40	79.80	77.33	67.92	213
13	75,533	165,171	–	1198.00	128.96	129.05	117.45	245
14	191,865	72,103	–	1364.85	127.17	128.17	115.90	346
15	35,170	237,017	1932.06	60.71	12.66	5.31	0.82	26
16	207,442	156,433	613.34	71.63	23.96	15.08	12.44	69
17	62,306	134,007	3211.47	294.02	22.60	22.05	10.22	78
18	58,427	135,252	2104.84	424.20	77.26	76.63	64.46	242
19	91,985	200,812	2484.05	433.97	98.17	97.36	85.71	241
20	242,644	163,590	1385.60	145.47	15.16	12.83	3.78	156
21	40,180	100,359	225.49	26.11	14.07	5.54	2.32	77
22	38,497	207,344	2340.13	534.32	202.15	199.53	190.24	465
23	180,834	83,150	–	–	1947.98	1948.15	1936.36	814
24	129,948	7003	2324.38	503.12	137.47	135.24	125.34	234
25	259,195	173,121	–	404.64	14.40	12.87	2.09	72
26	147,806	136,543	1541.13	337.06	69.30	63.93	55.57	371
27	179,874	57,536	–	–	1495.45	1495.73	1483.94	643
28	189,934	31,336	3327.36	416.25	29.56	25.34	17.02	169
29	138,263	253,856	7.58	1.02	12.69	0.64	0.04	11
30	246,144	166,336	625.33	32.43	13.43	6.84	1.00	65
31	25,610	143,842	627.76	88.28	16.76	10.71	4.21	86
32	228,779	167,251	–	372.19	19.49	18.60	8.03	162
33	78,936	34,136	–	535.16	39.74	39.86	27.58	111
34	124,173	138,439	2924.85	463.34	110.40	108.12	96.66	295
35	260,563	233,292	78.90	11.46	12.98	3.51	0.36	36
36	193,168	66,816	–	1127.19	95.75	94.74	83.40	280
37	29,432	29,834	2080.89	212.23	26.54	20.50	13.57	131
38	193,241	144,927	–	3586.53	208.97	209.09	196.63	787
39	161,522	171,446	14.66	2.75	11.37	0.80	0.02	1
40	176,910	109,129	3160.91	261.96	22.42	23.21	10.45	164
41	251,416	53,900	835.19	111.63	22.52	19.09	9.11	106
42	201,505	262,626	–	71.97	14.11	7.32	1.61	48
43	86,937	190,907	–	–	1389.32	1389.98	1377.66	632
44	35,252	18,638	20.72	2.16	12.66	1.53	0.03	4
45	92,562	65,120	–	452.98	35.23	33.85	23.14	202
46	230,423	2724	244.50	26.75	13.50	4.78	1.11	66
47	17,285	92,411	342.28	22.80	13.79	3.35	0.90	17
48	177,037	199,832	54.72	11.47	12.55	5.44	0.07	8
49	68,330	206,280	2130.97	665.56	115.93	114.12	102.93	270
50	61,414	50,367	147.27	19.25	13.89	11.32	1.47	50

Table A.8
Data from BAY map problems.

Prob.	Source	Goal	t_{h_0}	$t_{h_{cd}}$	$t_{h_{tc}}(o)$	$t_{h_{tc}}(b)$	t_{namoa}	$\lvert C^*\rvert$
1	217,950	116,998	–	–	1489.11	1489.03	1474.76	812
2	251,602	34,430	44.11	11.74	15.12	4.50	0.29	9
3	98,882	122,447	–	315.91	31.53	31.16	17.05	207
4	52,642	224,747	2725.53	202.15	17.23	11.38	3.59	43
5	139,849	299,950	504.02	159.59	15.41	12.53	1.06	46
6	227,292	149,896	3219.98	1302.36	180.20	180.24	166.23	374
7	184,605	142,630	80.72	7.19	14.34	4.65	0.98	27
8	61,991	285,138	–	1887.31	135.24	129.96	120.68	436
9	157,468	111,871	44.19	18.13	15.25	2.93	0.46	18
10	8057	53,146	29.85	2.73	14.02	1.18	0.04	9
11	63,481	317,962	3535.25	44.44	14.75	10.07	1.37	34
12	100,852	32,440	1139.09	234.63	16.21	15.48	2.74	45
13	88,253	223,600	1281.66	219.04	16.80	12.39	2.85	51
14	256,221	137,112	–	2899.57	328.21	329.05	314.01	445
15	296,236	87,228	0.25	0.13	14.58	0.12	0.00	1
16	79,044	118,604	38.27	12.16	13.84	2.71	0.03	3
17	101,931	260,608	2823.59	817.16	42.73	41.23	29.33	137
18	151,143	33,304	–	1528.02	137.36	137.21	123.21	353
19	252,289	254,898	2.20	0.95	14.55	1.27	0.04	9
20	168,761	275,306	1450.15	102.30	15.48	8.38	0.75	42
21	151,157	220,839	1965.79	464.95	77.32	76.09	62.89	203
22	50,213	139,654	3205.05	234.38	17.97	10.72	3.10	27
23	6574	319,340	3556.67	463.61	19.93	12.59	5.34	104
24	299,482	198,914	1487.94	29.64	14.28	8.76	0.27	16
25	319,552	169,293	438.47	46.49	15.33	10.27	0.63	25
26	214,934	40,905	534.58	182.76	38.23	35.90	22.60	61
27	41,820	197,615	754.82	263.92	30.99	24.91	16.85	175
28	88,417	292,062	5.65	2.23	15.27	1.40	0.02	2
29	190,939	185,572	1756.89	249.12	20.43	19.25	6.12	64
30	132,185	126,599	8.46	3.18	15.17	1.65	0.10	16
31	307,539	72,237	368.20	88.17	21.12	19.52	5.79	53
32	167,018	129,836	–	399.18	17.31	12.02	2.90	77
33	199,994	273,242	–	350.16	17.90	13.91	3.74	89
34	76,122	222,337	2068.02	203.48	17.89	12.13	4.03	35
35	60,060	15,213	47.29	7.60	15.46	5.48	0.77	42
36	35,111	110,152	–	1184.39	30.28	26.94	15.36	196
37	250,483	268,079	1285.62	249.11	18.30	13.77	3.00	145
38	177,519	38,511	1477.15	204.66	16.73	12.33	2.08	82
39	50,330	193,990	1040.35	329.55	30.82	23.37	16.44	144
40	132,178	166,402	–	432.11	31.56	30.44	16.94	133
41	193,656	269,296	1470.89	216.88	17.01	12.78	2.47	64
42	174,698	73,496	11.64	8.02	14.70	1.38	0.07	7
43	224,414	285,621	2796.62	1048.87	41.75	41.72	27.35	164
44	304,900	212,465	–	940.88	55.73	54.15	41.57	167
45	103,114	109,425	840.32	47.95	14.71	6.54	0.56	32
46	226,489	314,957	–	830.48	37.56	37.67	23.42	87
47	183,973	310,416	67.22	6.70	14.56	1.49	0.03	3
48	261,308	150,051	–	3329.43	409.74	409.49	395.36	499
49	70,155	317,172	–	97.18	16.86	13.39	2.68	75
50	118,218	48,399	717.68	222.91	15.90	16.43	1.27	53

and let k be a value such that,

$$a_k \leqslant d(n) < a_{k+1} \tag{2}$$

then,

$$N(n) = \sum_{j=0}^{k} N_j + \left\lceil \frac{d(n) - a_k}{L_{k+1}} \right\rceil \tag{3}$$

The corrected distance value $h_{cd}(n) = d(n) - e(n)$ is always a lower bound of the optimal *physical distance* calculated in the maps from n to the goal. A test over 10,000 randomly chosen pairs of nodes in each map revealed that the average error $e(n)$ subtracted in the heuristic calculations is just 0.67% of the great circle distance. In consequence, the estimate preserves most of the precision of great circle distances, while guaranteeing at the same time the admissible (optimistic) values needed by NAMOA*.

Notice that the a_i, N_i and L_i values can be easily precalculated for each map and are problem independent. Therefore, the calculation of heuristic estimates is quite efficient.

Regarding travel time, the optimistic estimate is to assume that the corrected great circle distance can be traversed at the maximum speed, i.e. a time estimate is calculated dividing by the largest factor, which is 1.0 in the DIMACS maps. Since $h_{cd}(n)$ is a lower bound on the physical distance of optimal paths from n to the goal, then $h_{cd}(n)/1.0$ is also a lower bound on travel time.

In consequence, the *corrected distances* multiobjective heuristic estimate is defined as $\vec{h}_{cd}(n) = (h_{cd}(n), h_{cd}(n))$.

4.2. TC heuristic

Tung and Chew (1992) proposed the precalculation of multiobjective heuristic estimates through search.

The TC heuristic is defined as $\vec{h}_{tc}(n) = (c_1^*(n), c_2^*(n))$, where $c_i^*(n)$ is the optimal cost of a path from n to the goal, considering only the ith cost component. These values are precalculated for each component by reversing all arcs in the graph, and applying Dijkstra's algorithm to find the shortest path from the goal node to all other nodes in the graph.

Notice that the TC heuristic estimates are trivially more accurate than those described in Section 4.1. However, TC estimates de-

Table A.9
Data from COL map problems.

Prob.	Source	Goal	t_{h_0}	$t_{h_{cd}}$	$t_{h_{tc}}(o)$	$t_{h_{tc}}(b)$	t_{namoa}	$\lvert C^*\rvert$
1	186,399	206,453	442.16	30.01	19.62	8.41	0.92	59
2	106,474	399,484	595.76	146.04	19.48	3.62	0.12	8
3	219,775	41,597	–	2166.09	23.30	12.33	3.98	134
4	240,731	182,571	–	–	471.60	470.38	451.03	655
5	417,012	345,347	–	3530.80	2555.49	2552.26	2536.21	544
6	218,133	101,499	31.42	8.99	18.89	1.79	0.50	33
7	82,209	119,557	–	2008.11	126.56	124.35	107.79	241
8	93,458	188,506	1598.14	202.76	26.64	23.17	7.08	106
9	342,828	9384	2124.14	505.43	71.07	64.82	52.32	324
10	173,475	148,741	–	–	783.07	782.61	764.60	1214
11	368,975	307,038	–	–	2149.43	2146.99	2130.25	1332
12	266,403	288,429	–	1084.79	247.95	239.11	229.84	477
13	39,969	357,717	–	856.05	58.79	55.57	40.22	506
14	311,623	141,945	–	–	2817.74	2817.19	2798.52	2028
15	192,685	230,403	–	526.20	36.01	31.58	16.36	221
16	242,487	387,257	61.86	33.47	19.14	4.12	0.46	29
17	260,867	187,773	3007.22	753.24	43.19	33.29	24.96	194
18	345,397	57,539	–	229.62	22.35	15.84	2.75	125
19	98,052	233,707	3206.37	650.37	37.10	27.72	17.92	81
20	16,974	272,085	–	–	–	–	–	–
21	373,200	393,176	–	1706.76	43.06	33.73	23.16	200
22	82,255	231,704	–	–	348.15	346.69	329.30	295
23	170,699	374,715	–	–	3570.71	3568.19	3551.32	1831
24	344,226	41,837	–	295.06	32.87	18.98	14.69	253
25	311,543	79,737	–	–	659.86	658.47	639.73	1119
26	233,022	342,755	–	–	313.39	313.15	293.43	715
27	287,214	273,946	–	2090.95	87.99	81.49	69.55	394
28	105,391	434,721	–	1546.68	295.80	287.43	275.79	407
29	261,648	309,214	–	2225.72	147.10	138.76	128.19	368
30	277,313	107,050	2764.01	420.47	23.47	21.63	4.70	70
31	38,771	199,544	2389.27	349.98	19.50	9.85	0.66	30
32	253,430	282,880	–	2680.23	41.35	38.61	22.66	178
33	175,642	183,753	121.57	36.06	19.31	8.86	0.11	6
34	313,912	290,255	2220.55	438.62	47.34	40.90	28.13	374
35	32,506	97,065	–	–	546.07	543.17	527.08	388
36	384,664	320,926	–	497.01	38.95	32.90	20.51	158
37	220,613	381,762	–	1665.09	23.98	15.88	5.00	169
38	276,063	158,585	146.34	54.17	29.91	26.57	10.75	89
39	104,682	393,819	–	–	1105.07	1102.20	1086.65	1272
40	6425	183,698	1012.25	542.74	37.47	24.73	17.67	164
41	185,170	419,257	1959.44	503.84	20.50	10.15	1.12	49
42	299,734	236,575	–	–	–	–	–	–
43	6364	172,422	–	907.95	23.35	14.01	4.83	105
44	108,080	330,816	882.63	355.54	34.19	22.60	15.38	119
45	60,885	371,450	1473.93	373.38	50.37	42.11	32.20	80
46	355,708	233,614	290.35	111.46	30.46	21.14	11.23	384
47	114,038	221,082	36.88	18.25	18.99	4.01	0.08	18
48	258,459	178,122	–	–	317.94	314.05	298.18	546
49	431,418	72,100	18.34	8.40	20.98	3.36	0.34	42
50	296,685	11,132	–	892.32	33.53	32.07	14.66	245

Table A.10
Data from FLA map problems.

Prob.	Source	Goal	t_{h_0}	$t_{h_{cd}}$	$t_{h_{tc}}(o)$	$t_{h_{tc}}(b)$	t_{namoa}	$\lvert C^*\rvert$
1	361,739	698,672	–	1103.68	53.47	21.61	7.95	186
2	686,602	27,291	–	–	–	–	–	–
3	546,667	1,044,042	–	2604.01	72.01	41.77	27.97	305
4	115,105	421,966	–	–	99.17	96.59	54.27	298
5	742,805	335,320	–	–	808.83	804.18	764.49	1187
6	88,673	333,047	–	–	1168.53	1163.91	1121.30	803
7	766,579	263,017	–	2567.95	149.87	125.11	103.42	280
8	28,100	848,660	–	–	835.81	835.32	790.38	779
9	134,765	11,866	–	–	290.13	279.88	243.91	612
10	158,576	949,455	2151.40	132.54	47.58	15.50	1.90	13
11	659,282	327,441	421.83	72.46	49.25	18.53	3.25	83
12	539,004	639,594	–	169.07	47.31	10.02	1.52	34
13	489,044	463,492	–	–	137.97	127.43	91.18	290
14	192,295	127,144	–	–	–	–	–	–
15	481,860	1,046,443	1912.12	456.51	56.81	19.87	10.29	145
16	273,776	154,436	3375.94	721.89	62.01	50.86	13.97	117
17	946,451	513,773	–	2023.97	89.21	62.32	43.00	269
18	90,921	359,195	–	–	1162.02	1160.10	1115.85	848
19	783,218	996,886	–	–	406.23	388.91	360.44	517
20	1,052,751	190,896	–	–	–	–	–	–
21	646,797	149,214	–	3321.75	103.10	104.12	58.34	182
22	398,569	982,263	739.69	189.77	52.85	18.25	5.94	222
23	809,772	870,827	–	–	887.27	871.37	843.37	605
24	635,036	38,956	–	–	–	–	–	–
25	234,560	955,775	2275.86	210.93	50.14	23.02	2.94	56
26	274,945	143,720	–	–	–	–	–	–
27	716,892	344,531	–	–	761.94	756.30	715.94	1063
28	516,174	154,020	–	1419.68	88.40	68.99	41.82	247
29	129,998	118,211	–	536.97	60.84	32.44	16.36	109
30	905,861	756,883	56.46	53.19	50.82	9.08	3.87	144
31	41,614	404,340	296.99	42.90	49.53	5.53	1.60	9
32	933,700	561,390	–	3566.71	48.41	28.18	1.42	8
33	237,886	310,889	–	–	877.19	864.33	829.91	824
34	257,739	652,062	69.65	34.88	48.35	2.86	1.53	21
35	478,200	1,062,969	1885.46	855.62	172.31	134.38	125.32	656
36	173,720	246,425	–	–	51.84	22.18	7.63	161
37	43,974	803,673	–	3449.18	75.92	56.61	29.53	214
38	382,275	1,044,332	–	1715.21	86.79	68.37	41.86	310
39	462,808	85,391	2012.71	148.33	46.82	13.72	2.46	67
40	643,063	593,489	–	–	–	–	–	–
41	310,505	612,278	–	–	3248.15	3237.24	3203.14	1168
42	818,016	667,330	–	2327.15	101.68	74.78	55.41	264
43	257,389	17,759	–	–	915.11	916.32	870.24	570
44	16,738	751,658	–	–	276.51	269.02	232.90	371
45	401,809	616,933	–	–	3341.79	3341.38	3298.90	1217
46	364,903	404,709	–	3411.00	135.65	119.92	89.59	188
47	624,617	364,615	–	–	–	–	–	–
48	472,495	193,187	3353.50	277.91	52.87	26.99	9.56	94
49	131,865	294,161	416.87	67.16	48.70	10.91	3.87	120
50	363,001	263,258	–	2011.66	221.10	195.72	175.66	216

pend on the goal node and generally need to be precalculated for each problem instance. The calculation procedure is also much more costly computationally.

4.3. Bounded calculation for the TC heuristic

The preprocessing described in the previous section calculates estimates for *all* nodes in the graph. It is obvious that many problem instances require the examination of fewer nodes.

Let C^* be the set of nondominated solution costs for a given problem. It can be shown that NAMOA* will never consider for expansion paths whose estimates are dominated by some vector in C^* (Mandow & Pérez de la Cruz, 2010) (Lemma 4.5). This property can be conveniently used to limit the number of nodes examined in the precalculation of the heuristic values. Let us denote by c_1^*, c_2^* the optimal solutions for the single objective under consideration. Let c_2' be the minimum value of the second objective among all solution paths attaining c_1^*, and c_1' the minimum value in the first objective among those paths attaining c_2^* in the second one.

Fig. 2 displays these values in a biobjective cost space. All nondominated solution costs to a problem $\vec{c}^* \in C^*$ lie by definition in rectangle A defined by the two extreme points[3] (c_1^*, c_2') and (c_1', c_2^*).

Lemma 4.5 (Mandow & Pérez de la Cruz, 2010) implies that NAMOA* will never consider for expansion paths in rectangle C, with costs dominated by (c_1', c_2'), since these can never lead to nondominated solutions. Therefore, heuristic estimates for nodes n with optimal costs $c_1^*(n) > c_1'$ and $c_2^*(n) > c_2'$ do not need to be calculated (i.e. they could be set to infinity).

The following three-stage calculation procedure is proposed,

1. Reverse all arcs in the graph, and apply a modified Dijkstra's algorithm to find the shortest path from the goal node to other nodes in the graph. The modified Dijkstra's algorithm minimizes c_1 values, but in case of ties, prefers paths with smaller value of c_2, i.e. optimizes using a total lexicographic order[4] defined by $(c_1(n),c_2(n))$. Pause search as soon as the start node is selected for expansion. This will provide the optimal lexicographic cost (c_1^*, c_2').
2. Reverse all arcs in the graph, and apply a modified Dijkstra's algorithm to find the shortest path from the goal node to other nodes in the graph using a lexicographic order defined by $(c_2(n),c_1(n))$. Stop search as soon as a node n selected for expansion satisfies $c_2(n) > c_2'$. Notice that the start node will be selected at some time during this search, and labeled with the optimal lexicographic cost (c_1', c_2^*).
3. Resume the search paused in stage 1, and stop as soon as a node n selected for expansion satisfies $c_1(n) > c_1'$.

Notice that this procedure calculates exactly the same heuristic values of the TC heuristic, except for those nodes that will never be considered by NAMOA* (i.e. those with estimates in rectangle B of Fig. 2). These missing values can be set to infinity.

5. Experimental results

The performance of NAMOA* with different heuristic functions has been analyzed over 200 random problems in four road maps of increasing size. Each problem instance was solved 5 times with each heuristic. Average time values are presented in the results. The heuristics under evaluation are blind search $\left(\vec{h}_0\right)$, corrected great circle distance $\left(\vec{h}_{cd}\right)$, and Tung and Chew heuristic $\left(\vec{h}_{tc}\right)$ with both the original method and the improved calculation method. Time values presented for NAMOA* with $\vec{h}_{tc}$ include heuristic precalculation time plus multiobjective search time. For each problem instance the Tung and Chew heuristic was precalculated 10 times using the method of choice (see Sections 4.2 and 4.3). Average precalculation values are added in the results to NAMOA* as heuristic precalculation time when these heuristics were applied.

The algorithms were implemented in ANSI Common Lisp using LispWorks 6.0 Enterprise 64 bits, and run on a Sun Fire X4140 server with 2 six-core AMD Opteron 2435 @ 2.60 GHz processors and 64 Gb of DDR2 RAM, under Windows Server 2008 R2 Enterprise (64-bits). Each problem instance was solved using an individual process with a single thread. Open labels at each node were kept in lexicographically ordered lists. The best representative of each node is inserted in the *OPEN* list, implemented as a binary heap.

The time requirements of NAMOA* for the NY, BAY, COL and FLA maps (see Table 1) are presented in Figs. 3–6 respectively. The ordinate axis x shows problem indexes for each instance in the 50 problem set for each map. Instances are ordered according to increasing time taken by NAMOA* with heuristic $\vec{h}_{tc}$ calculated using the bounded method. The value for abscissa y (time in seconds) is shown in a logarithmic scale. Source and destination nodes

Table 2
Number of problems that could not be solved in 1 h.

Heuristic	NY	BAY	COL	FLA
h_0	17	12	30	37
h_{cd}	3	1	13	23
h_{tc}	0	0	2	7

Table 3
Number of successful executions for instances that exceeded at least once, but not always, the time limit.

ID #	NY38	BAY6	BAY23	BAY48	COL5	COL23	FLA32	FLA41,45	FLA48
h_0	0	3	2	0	0	0	0	0	4
h_{cd}	1	5	5	1	1	0	3	0	5
h_{tc}	5	5	5	5	5	1	5	3	5

Table 4
Speed up with respect to h_0.

Map	Heuristic		
	h_{cd}	h_{tc} (*orig*)	h_{tc} (*bounded*)
NY	6.20	29.19	33.50
BAY	6.07	49.77	74.37
COL	4.40	40.13	60.52
FLA	5.41	18.18	45.05
Overall	5.52	34.32	53.36

Table 5
Speedup with respect to h_{cd}.

Map	NAMOA* with	
	h_{tc} (*orig*)	h_{tc} (*bounded*)
NY	5.59	5.94
BAY	10.53	12.44
COL	14.24	17.09
FLA	14.43	22.48
Overall	11.20	14.49

[3] Note that the TC heuristic estimate for the start node is (c_1^*, c_2^*), which trivially dominates all points in rectangle A.

[4] A detailed analysis of such algorithm can be found elsewhere (Mandow & Pérez de la Cruz, 2003).

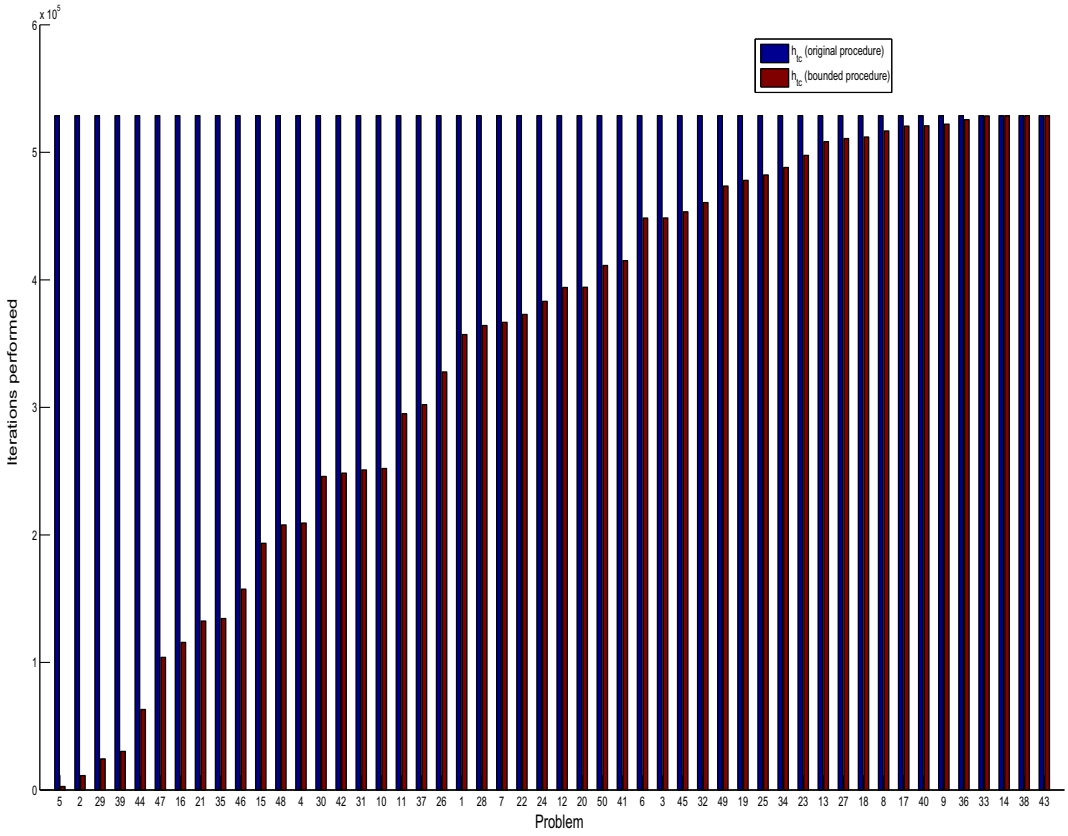

Fig. 7. Iterations performed by the original and bounded methods for the TC heuristic on NY problem instances.

for each problem instance, along with time values for each heuristic are presented also in Tables A.7–A.10. Additional information, like the number of distinct nondominated costs for each instance $|C^*|$ is also presented in these tables.

For practical reasons, instances were solved each time with a 1 h time limit. For certain instances, none of the five executions performed with a given heuristic where able to solve the problem within the time limit. In such cases, values are not displayed in the figures for that particular heuristic. Table 2 shows the number of problem instances that were unsolvable within the time limit for each map and heuristic. In a few instance-heuristic combinations only some of the five executions exceeded the time limit. Therefore, these seem to require an average time very close to the limit. Averages presented for these instance-heuristic combinations take into account only successful executions. Table 3 lists these particular instances, and how many successful executions were run within the time limit for each heuristic.

6. Discussion

Blind search vs. heuristic search. The results presented in Section 5 show that the heuristic approaches ($\vec{h}_{cd}, \vec{h}_{tc}$) clearly outperform blind search ($\vec{h}_0$), both in the number of instances solved (see Table 2), and in run time (see Figs. 3–6). Blind search was not able to solve the full instance set for any of the maps within the time limit, and performed rather poorly failing in 30 and 37 cases out of 50 in the COL and FLA maps respectively. Regarding run time, blind search was beaten in all instances by $\vec{h}_{cd}$ and $\vec{h}_{tc}$ with the improved calculation method. However, blind search was able to beat $\vec{h}_{tc}$ with the original method in 9 of the easiest instances (3 in NY, 5 in BAY, and 1 in COL). This was due to the constant overhead in the heuristic precalculation for this method, regardless of problem difficulty.

For the subset of problem instances that could be solved by blind search,[5] the overall average speedup of heuristic search was 5.52 for $\vec{h}_{cd}$, and 34.32 and 53.36 for $\vec{h}_{tc}$ with the original and improved methods respectively. In the case of $\vec{h}_{tc}$ the speedup was much larger for BAY and COL maps than for the NY and FLA ones (see Table 4).

Performance of heuristic search. Regarding the relative performance of the heuristic approaches, Table 2 shows that $\vec{h}_{cd}$, although consistently better than $\vec{h}_0$, is clearly outperformed by $\vec{h}_{tc}$ in terms of the number of solved instances. In fact, $\vec{h}_{tc}$ could solve the complete sets for NY and BAY within the time limit, and failed only for 2 and 7 problems in COL and FLA respectively.

Time performance with the improved method for $\vec{h}_{tc}$ outperformed $\vec{h}_{cd}$ in all cases, except for two simple instances (NY 5 and BAY 19), where the overhead of precalculation did not compensate for the very small search effort needed. In fact, even for these instances, NAMOA* performed less iterations with $\vec{h}_{tc}$ than with $\vec{h}_{cd}$. However, $\vec{h}_{cd}$ was able to beat $\vec{h}_{tc}$ with the original calculation method in 22 of the simpler problem instances. Once again, this was due to the constant overhead in the heuristic precalculation for this method, regardless of problem difficulty.

For the set of problem instances[6] that could be solved by $\vec{h}_{cd}$, the overall average speedup of $\vec{h}_{tc}$ was 11.20 and 14.49 with the original and improved methods respectively (see Table 5). More precisely, the speedup of $\vec{h}_{tc}$ (improved method) over $\vec{h}_{cd}$ is of 5.94, 12.44, 17.09, and 22.48 for the NY, BAY, COL, and FLA maps, indicating that the Tung–Chew heuristic becomes increasingly better than corrected great circle distance for larger maps. This is probably related not only

[5] Instances that could not be solved in all five runs within the 1 h time limit by $\vec{h}_0$ are not considered.

[6] Instances that could not be solved in all five runs within the 1 h time limit by $\vec{h}_{cd}$ are not considered.

Table 6
Run time in seconds for problem instances run without time limit with bounded $\vec{h}_{tc}$.

ID #	COL20	COL42	FLA2	FLA14	FLA20	FLA24	FLA26	FLA40	FLA47
h_{tc}	4026	12,783	12,578	140,212	58,246	78,422	13,531	144,834	19,504

with graph size, but with the fact that the larger graphs have also larger physical extension.

These results confirm that more accurate heuristic estimates result in improved time performance in NAMOA*, since $\vec{h}_{tc}$ is trivially more informed than $\vec{h}_{cd}$.

Effectiveness of the Tung and Chew heuristic. The results presented in Section 5 clearly indicate that $\vec{h}_{tc}$ is by far the best heuristic among the alternatives considered. Search time taken by NAMOA* with this heuristic is much smaller than with h_{cd}, the second best alternative (see Tables A.7–A.10).

The experiments also clarify the adequacy of heuristics precalculated through search in multiobjective problems. Even when the time taken in the precalculations is added to the search time of NAMOA*, this alternative remains very competitive. The original calculation method calls for a one-to-all search of shortest paths for each objective under consideration. This implies an overhead of about 10 s in the NY and BAY maps, and of 11 and 14 s in COL and FLA respectively. However, the formal properties of NAMOA* allow us to bound the number of nodes whose heuristic needs to be precalculated. With this bounded calculation method h_{tc} was beaten by h_{cd} only in two simple cases, NY 5 and BAY 19. In NY 5 the precalculations took 2729 iterations, while NAMOA* took only 21 and 61 iterations with h_{tc} and h_{cd} respectively. In BAY 19, the precalculations took 40,790 iterations, while NAMOA* performed 613 and 6446 iterations with h_{tc} and h_{cd} respectively.

Fig. 7 shows the total number of iterations performed by the original TC heuristic calculation and the new bounded method in the case of NY map for the 50 randomly chosen problems. Similar values are obtained for the other maps. The original TC method made exactly the same effort for all problems (i.e. all the nodes in the graph were explored twice). The bounded method can improve efficiency in many cases, though in difficult problems, where source and destination nodes are far apart, it is necessary to calculate heuristic values for all nodes in the graph regardless of the calculation method.

Among the alternatives considered in this article, the h_{tc} heuristic with the bounded calculation method is clearly the option of choice for multiobjective search in road maps.

Difficult problem instances. A set of nine problem instances could not be solved within the time limit regardless of the heuristic function. These problem instances were solved once with the $\vec{h}_{tc}$ heuristic with bounded calculation, which is identified in the analyses as the most effective alternative. All of them could be solved with the available resources. Table 6 shows the time taken to solve each instance. The hardest one, FLA 40, was solved in about 40 h and 14 min.

7. Conclusions and future work

This article reports the successful application of exact multiobjective techniques to search in large size realistic road maps. Road map data were obtained from the 9th DIMACS implementation challenge on shortest paths.

The NAMOA* multiobjective heuristic search algorithm has been tested with a heuristic precalculated with search (TC), a classical distance heuristic (corrected great circle distance) and without heuristic information (blind search). In addition to the original calculation method for the TC heuristic, a new bounded calculation method is proposed. This method takes advantage of formal properties of the NAMOA* algorithm to reduce the precalculation effort. All the heuristics and precalculation methods considered were run on a set of 200 random problems defined over four different maps.

Results show that the combination of NAMOA* with the TC heuristic is a competitive approach in the solution of biobjective search problems. Furthermore, the time devoted to heuristic precalculations has been found to pay off, greatly reducing multiobjective search effort. Additionally, the new bounded calculation method for the TC heuristic has been found to be more efficient in practice than the original method, significantly reducing precalculations in many problem instances.

The TC heuristic with the new bounded calculation method was found to clearly outperform a classical distance heuristic and blind search, both in the number of solved problems and in time performance. The overall speedup of the TC heuristic with bounded calculation in the experiments is 53.56 and 14.49 when compared to blind search and the distance heuristic respectively.

The approach described in this article could solve random problem instances in realistic road maps of up to 1,070,376 nodes and 2,712,798 arcs. Most instances could be solved within a 1 h time limit and the hardest one was solved in over 40 h, which is reasonable for many off-line route planning applications.

Future work includes the evaluation of this scheme in other problem domains, as well as the consideration of further improvements in the calculation of the heuristic values. A number of techniques have been proven very effective in real-time single-objective route planning applications, like search in car navigation systems. The extension of these techniques to multiobjective settings is an important area of research. However, the results presented in this article suggest that techniques based in extensive blind multiobjective precalculations are likely to experience practical difficulties in large maps, as already noted by Delling and Wagner (2009).

The use of precalculated heuristics in multiobjective search is a rather unexplored area that deserves formal and practical developments. The application described in this article could clearly benefit from more informed precalculated heuristics. The impact of the TC heuristic in other practical problem domains should also be investigated.

Finally, the application of other multicriteria approaches could also be investigated. Specialized multicriteria algorithms that search for compromise solutions (Galand & Perny, 2006) or apply OWA based search (Galand & Spanjaard, 2007) do not need to compute the whole Pareto-optimal set. If adequate heuristic functions could be devised for these methods they could also be efficient alternatives in specialized applications.

Appendix A. Problem data

Tables A.7–A.10 display the experimental data shown in Figs. 3–6 respectively. Additional information is presented for each problem instance, like source and destination identifiers in the DIMACS challenge maps, multiobjective search time of NAMOA* (without precalculation time) for the h_{tc} heuristic, and the number of different nondominated solution cost vectors $|C^*|$. Average search times for the different heuristics are in seconds. Times for problem instances that could not complete all 5 runs within the 1 h time limit are averaged only over successful runs and are marked with underlined text.

References

Bauer, R., Columbus, T., Katz, B., Krug, M., & Wagner, D. (2010). Preprocessing speed-up techniques is hard. In *CIAC* (pp. 359–370).

Caramia, M., Giordani, S., & Iovanella, A. (2010). On the selection of k routes in multiobjective hazmat route planning. *IMA Journal of Management Mathematics, 21*, 239–251.

Culberson, J. C., & Schaeffer, J. (1998). Pattern databases. *Computational Intelligence, 14*, 318–334.

Dechter, R., & Pearl, J. (1985). Generalized best-first search strategies and the optimality of A*. *Journal of the ACM, 32*, 505–536.

Delling, D., & Wagner, D. (2009). Pareto paths with SHARC. In *SEA* (pp. 125–136).

Delling, D., Sanders, P., Schultes, D., & Wagner, D. (2009). Engineering route planning algorithms. In *Algorithmics. LNCS* (Vol. 5515, pp. 117–139). Springer.

Dell'Olmo, P., Gentili, M., & Scozzari, A. (2005). On finding dissimilar Pareto-optimal paths. *European Journal of Operational Research, 162*, 70–82.

Dijkstra, E. W. (1959). A note on two problems in connexion with graphs. *Numerische Mathematik, 1*, 269–271.

Erkut, E., Tjandra, S. A., & Verter, V. (2007). Hazardous materials transportation. In C. Barnhart & G. Laporte (Eds.). *Handbook in OR and MS* (Vol. 14, pp. 539–621). Germany: Elsevier.

Galand, L., & Perny, P. (2006). Search for compromise solutions in multiobjective state space graphs. In *Proc. of the XVII european conference on artificial intelligence (ECAI'2006)* (pp. 93–97).

Galand, L., & Spanjaard, O. 2007. Owa-based search in state space graphs with multiple cost functions. In *FLAIRS conference 2007* (pp. 86–91).

Goldberg, A.V., & Harrelson, C. (2005). Computing the shortest path: A* search meets graph theory. In *SODA'05 proceedings of the sixteenth annual ACM-SIAM symposium on discrete algorithms* (pp. 156–165).

Hansen, P. (1979). Bicriterion path problems. *Lecture notes in economics and mathematical systems* (Vol. 177, pp. 109–127). Springer.

Hansson, O., Mayer, A., & Valtorta, M. (1992). A new result on the complexity of heuristic estimates for the A* algorithm. *Artificial Intelligence, 55*, 129–143.

Hart, P., Nilsson, N., & Raphael, B. (1968). A formal basis for the heuristic determination of minimum cost paths. *IEEE Transactions on Systems Science and Cybernetics SSC-4, 2*, 100–107.

Holte, R. C., Perez, M. B., Zimmer, R., & MacDonald, J. (1996). Hierarchical A*: Searching abstraction hierarchies efficiently. In *Proc. thirteenth national conference on artificial intelligence (AAAI-96)* (pp. 530–535).

Holte, R. C., Grajkowski, J., & Tanner, B. (2005). Hierarchical heuristic search revisited. In J. Zucker, & L. Saitta (Eds.), *SARA 2005, LNAI* (Vol. 3607, pp. 121–133).

Jozefowiez, N., Semet, F., & Talbi, E.-G. (2008). Multi-objective vehicle routing problems. *European Journal of Operational Research, 189*, 293–309.

Kanoh, H. (2007). Dynamic route planning for car navigation systems using virus genetic algorithms. *International Journal of Knowledge-based and Intelligent Engineering Systems, 11*, 65–78.

Kanoh, H., & Hara, K. (2008). Hybrid genetic algorithm for dynamic multi-objective route planning with predicted traffic in a real-world road network. In *GECCO'08* (pp. 657–664).

Kim, B.-K., Jo, J.-B., Kim, J.-R., & Gen, M. (2009). Optimal route search in car navigation systems by multi-objective genetic algorithms. *International Journal of Information Systems for Logistics and Management, 4*, 9–18.

Machuca, E., Mandow, L., & Pérez de la Cruz, J. L. (2009). An evaluation of heuristic functions for bicriterion shortest path problems. In *Proc. of the 14th Portuguese conference on artificial intelligence (EPIA 2009)* (pp. 205–216).

Mandow, L., & Pérez de la Cruz, J. L. (2009). A memory-efficient search strategy for multiobjective shortest path problems. In B. Mertsching, M. Hund, & Z. Aziz (Eds.), *KI 2009: Advances in artificial intelligence, LNAI* (Vol. 5803, pp. 25–32).

Mandow, L., & Pérez de la Cruz, J. L. (2003). Multicriteria heuristic search. *European Journal of Operational Research, 150*, 253–280.

Mandow, L., & Pérez de la Cruz, J. L. (2010). Multiobjective A* search with consistent heuristics. *Journal of the ACM, 57*, 27:1–25.

Martins, E. (1984). On a multicriteria shortest path problem. *European Journal of Operational Research, 16*, 236–245.

Müller-Hannemann, M., & Weihe, K. (2006). On the cardinality of the Pareto set in bicriteria shortest path problems. *Annals OR, 147*, 269–286.

Nannicini, G., Delling, D., Liberti, L., & Schultes, D. (2008). Bidirectional A* search for time-dependent fast paths. In *WEA* (pp. 334–346).

Pearl, J. (1984). *Heuristics*. Reading, Massachusetts: Addison-Wesley.

Raith, A., & Ehrgott, M. (2009). A comparison of solution strategies for biobjective shortest path problems. *Computers and Operations Research, 36*, 1299–1331.

Schultes, D. (2008). *Route planning in road networks*. Ph.D. thesis Fakultät für Informatik der Universität Fridericiana zu Karlsruhe (TH).

Stewart, B. S., & White, C. C. (1991). Multiobjective A*. *Journal of the ACM, 38*, 775–814.

Tung, C. T., & Chew, K. L. (1992). A multicriteria Pareto-optimal path algorithm. *European Journal of Operational Research, 62*, 203–209.

Valtorta, M. (1984). A result on the computational complexity of heuristic estimates for the A* algorithm. *Information Science, 34*, 48–59.